

**Título :**

**“Processo de introdução de melhorias e de correcções de erros de software em Sistemas de Telecomunicações”.**

Autores :           Eng° Sousa Oliveira.  
                          Eng° Miguel Prudêncio.

SIEMENS, S.A., Dept. Comutação e Software.  
Av. Almirante Reis, nº 65       1150 LISBOA.  
Telef: (01) 350 2000       Fax: (01) 350 2424

**Resumo**

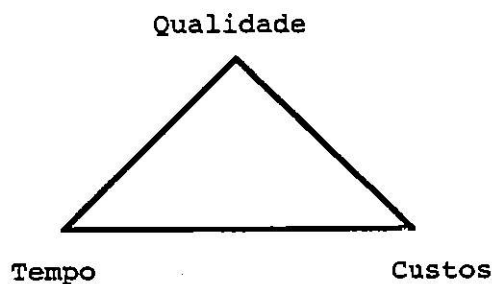
No contexto de melhoria contínua, esta comunicação aborda o tema das alterações introduzidas no software de sistemas hierarquizados, isto é, divididos em subsistemas e módulos, para a implementação de melhorias ou correcção de erros, garantindo a compatibilidade e a divulgação coerente das sucessivas versões dos diferentes módulos que constituem o sistema.

Esta comunicação onde serão ainda referidas ferramentas de apoio para controlo da rastreabilidade e das responsabilidades inerentes ao processo, está organizada em três capítulos, o primeiro constituindo uma introdução global e os dois seguintes abordando, respectivamente, os relatórios de falhas e a implementação das correcções.

Conclusões, lista de abreviaturas e definições e bibliografia completam a comunicação.

**1. Introdução**

Os mercados abertos e a conseqüente concorrência permanente levam a manter sempre presente o triângulo:



que, por outras palavras, significa:

- evolução tecnológica e de facilidades,
- melhoria de processos,
- eliminação de erros.

Esta comunicação aborda em detalhe a metodologia do tratamento de erros desde o seu levantamento (relatório de falha) seja ele interno ou feito pelo cliente, até à sua solução e implementação no campo.

Convém aqui referir que se trata de software existente em centrais telefónicas ou outros sistemas de telecomunicações e que, portanto, se encontra espalhado por todo o país, eventualmente em diferentes fases de evolução, isto é, em diferentes versões.

Igualmente convém separar os erros de software das avarias de hardware, visto serem tratados de formas diferentes:

- Os erros de software dão origem a correcções no código.
- As avarias de hardware são resolvidas por substituição e/ou reparação do dispositivo avariado. O registo e tratamento estatístico destas avarias permite evidenciar eventuais pontos fracos e a consequente tomada das acções correctivas e preventivas. O registo das avarias referenciado ao dispositivo e ao componente assim como a conservação do respectivo histórico permitem a rastreabilidade do processo e a consequente evidência de falhas ocasionais ou sistemáticas quer ao nível do processo quer ao nível do produto.

Os erros de software levantados pelos relatórios de falha (*Fault reports*) são corrigidos:

- através do código fonte implicando nova compilação e “linkagem”
- através de alterações introduzidas directamente no código máquina, tomando, neste caso, o nome de *patch*.

Quando o software de aplicação (APS) ainda se encontra numa fase inicial de teste, os erros detectados nessa fase são, regra geral, corrigidos na fonte e, posteriormente, feita nova produção (compilação) dessa versão de software.

Quando um APS se encontra em fase final de testes e especialmente se a versão de software já se encontra instalada no campo, as correcções de erros são feitas por *patches* físicas.

Estas *patches* são introduzidas no APS através de ficheiros que executam vários comandos.

A nossa comunicação vai abordar apenas este segundo caso.

Uma vez criado o *patch*, este é testado inicialmente em sistemas simulados e depois nas nossas centrais de teste equipadas com o hardware e o software que lhes permite criar condições iguais às verificadas no campo quando o erro ocorreu.

Após aprovação deste pelo cliente é feita a introdução do *patch* em todas as centrais onde aquela versão de software esteja implementada. Parte-se do princípio que um erro no código do software está presente em todas as centrais onde a versão do módulo de software afectado estiver instalada, independentemente de ter sido detectado ou não.

A evolução do software EWSD, o processo de teste apertado e um acompanhamento permanente nos primeiros tempos após a entrega da versão piloto ao cliente, assim como as condições específicas de cada central dependentes da sua localização na rede, permitem afirmar que, normalmente, novos erros só aparecem em situações muito especiais em que se dá a coincidência de vários factores, o que justifica que uma determinada situação de erro só ocorre numa central e não em todas.

Igualmente, o processo de controlo dos *patches* garante a sua inclusão em futuros melhoramentos dos respectivos módulos assim como a sua divulgação a nível mundial.

Resumindo, a eliminação dos erros de software é feita através da aplicação de dois processos paralelos e interligados:

- Processo “Relatório de Falha” que contempla o registo de todos os dados e fases que conduzem à solução do problema.
- Processo *Patch* que contempla a alteração do programa (código) e a sua implementação no campo.

Por outras palavras o processo *patch* resolve o problema eliminando a causa do erro e o processo **Relatório de Falha** regista todas os elementos e passos que permitiram a solução do problema.

## **2. Princípios genéricos do processamento do Relatório de Falha (*Fault Report*)**

### **2.1 Objectivos**

A normalização de conceitos relacionada com o processamento de *Fault reports*, permitirá resolver os erros de software encontrados de forma mais rápida.  
A qualidade das correcções é pois garantida pela normalização de conceitos.

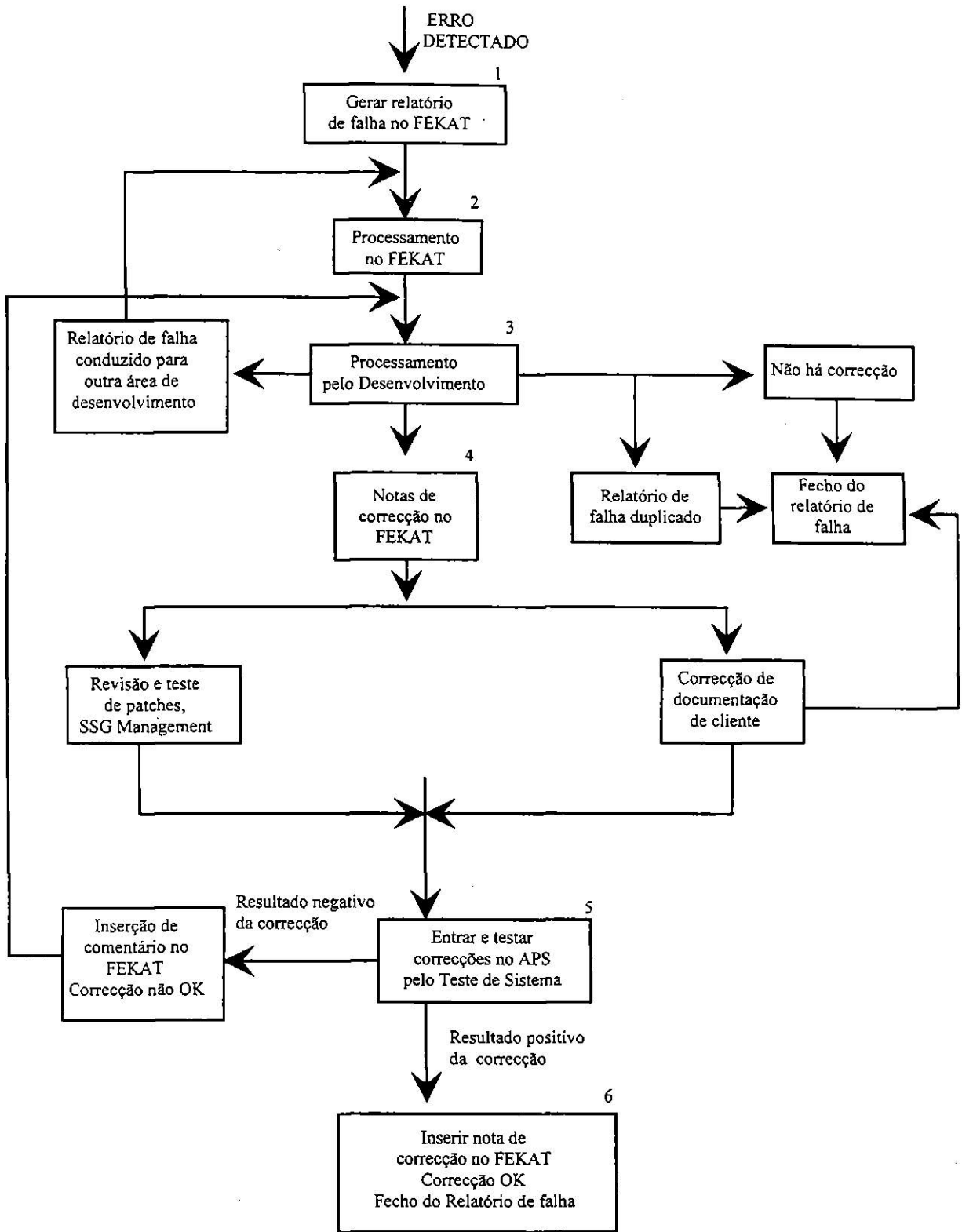
### **2.2 Processo**

(ver fluxograma Relatórios de Falha)

#### **2.2.1 Fluxograma**

Ver figura 1

As alíneas seguintes descrevem as principais fases do fluxograma. Os números que acompanham os títulos referem as respectivas fases no fluxograma.



**Fig.1 - Fluxograma de processamento de relatórios de falha**

### 2.2.2 Inserção (1)

Introdução de elementos descrevendo o erro, no que se refere ao projecto envolvido, sub sistema, descrição pormenorizada, início do processo, organização responsável, bem como a prioridade de resolução - técnica e de cliente .

### 2.2.3 Pré-análise (2)

Localização do erro dentro da organização do sistema que, devido à estrutura modular do software, permite associar aos diferentes grupos responsáveis pela sua correcção.

### 2.2.4 Análise (3)

Identificação de falha que causa o problema.  
Planeamento de acções correctivas.

### 2.2.5 Correcção (4)

Desenvolvimento, e teste das medidas correctivas.

### 2.2.6 Verificação (5)

Teste pelo nível superior, garantindo a funcionalidade do sistema.  
Libertação das correcções, isto é, a autorização para colocar no campo

### 2.2.7 Fecho (6)

Teste da correcção já no ambiente de funcionamento da correcção.  
Ex: 'APS de campo'.  
Comentário final do *Fault Report*.

## 2.3 **Controlo**

Os *Fault Reports* são processados por várias entidades, logo, particular importância deve ser dada ao controlo de processo.

Dentro do processo, uma determinada tarefa é iniciada logo que esteja disponível a informação da tarefa anterior. Para cada tarefa deve estar indicado o responsável e o objectivo.

Quando uma tarefa está terminada, devem ser claros os resultados do processo nessa fase, bem como a próxima tarefa. Normalmente, no final de cada tarefa, os resultados devem ser determinados. Modificações posteriores não são permitidas. Em caso de absoluta necessidade, dever-se-á abrir novo processo.

Em qualquer altura do processo deve ser possível inquirir o estado de um *Fault Report*, e a seguinte informação deve estar sempre disponível:

- tarefas em aberto;
- tarefas completadas;
- organização responsável;
- resultados correntes.

Durante o processo, a sequência de tarefas tem que ser transparente para qualquer das entidades envolvidas.

A responsabilidade para cada uma das tarefas tem que ser clara. A organização prevê a coordenação de todas as actividades relacionadas com um *Fault Report*.

É também possível monitorizar grupos de *Fault Reports* por sistema, projecto ou produto, bem como a sua observação individual.

Os tempos de reacção, visto normalmente o *Fault Report* passar por várias organizações, são também controlados.

## 2.4 Ferramentas de apoio

Para implementar os princípios atrás enunciados, o procedimento de *Fault Report* é apoiado, em ferramentas.

- Para processamento local. - FMERF/PROCONS;
- Para processamento central. - FEKAT.

Estas ferramentas usam interfaces com outras ferramentas, em particular com a CM - *Configuration Management* e *SSG Management*.

O processamento de um *Fault Report* envolve as fases enunciadas em 2.2.

## 2.5 Entidades

As entidades envolvidas durante o processamento de *Fault Reports* são:

- **Apoio ao Cliente**

Esta entidade existe exclusivamente para *Fault Reports* emitidos pelo cliente. Uma organização local ou central pode ser envolvida dependendo da estrutura da organização.

- **Apoio ao Relatório de Falha**

Esta entidade controla a distribuição, monitorização e apoio ao processamento de *Fault Reports*. Esta tarefa é assumida pelo responsável do sistema com falha e pode ser alterada.

- **Equipa de Desenvolvimento**

Esta entidade é responsável pela análise do erro e, se necessário, a sua correcção no respectivo Sub-Sistema.

- **Equipa de Teste**

Esta entidade assegura a funcionalidade dos Sub-Sistemas corrigidos em conjugação com outras funcionalidades de sistema. Isto inclui introdução de correcções, verificação e análise de falhas e verificação de correcções.

### **3. Principios genéricos do processamento de 'Patches'**

#### **3.1 Objectivos**

Os *patches* são correcções de código-objecto de um software de aplicação (APS). Um *patch* torna possível a correcção de erros durante a operação de um sistema sem interrupção das suas funcionalidades normais. Descrições de *patches* fazem parte da documentação oficial entregue ao cliente, sendo sujeitas a um apertado controlo de qualidade.

#### **3.2 Fases de processamento de Patches**

(ver fluxograma *Patches*)

##### **3.2.1 Fluxograma**

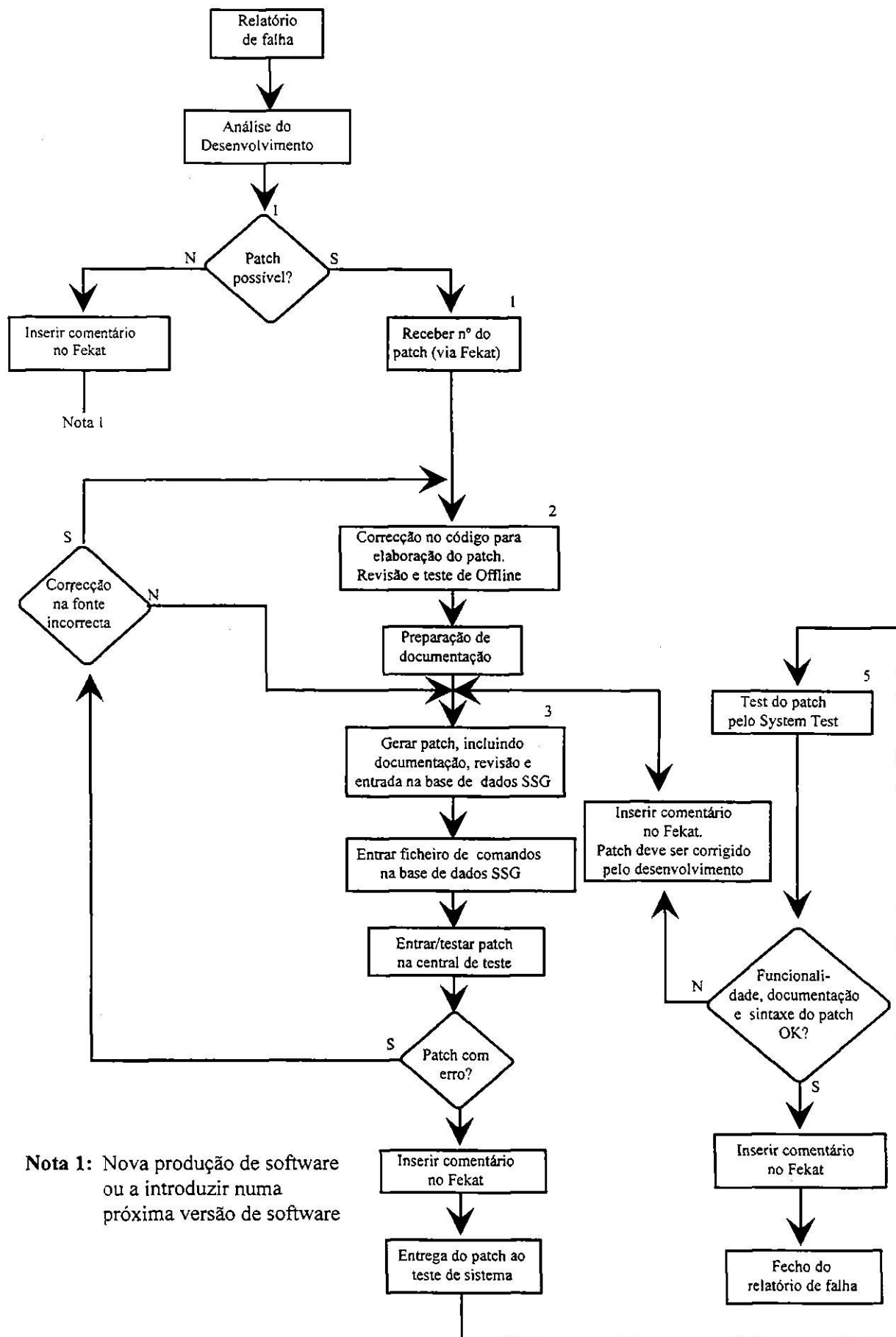
Ver figura 2

##### **3.2.2 Geração do Patch (1)**

Depois da análise de um relatório de falha, e caso o erro deva ser corrigido através de um *patch*, deverá a área de desenvolvimento verificar o troço de código onde o mesmo irá ser inserido para determinar se tecnicamente é possível a sua introdução. Nessa verificação deverão ser identificadas as diferenças de códigos.

##### **3.2.3 Teste em ambiente off-line (2)**

O *patch* é testado pela equipa de desenvolvimento num simulador ou num ambiente de teste autónomo.



**Fig. 2 - Fluxograma do processamento de Patches**



### 3.2.4 Documentação do Patch na base de dados SSG (3)

A documentação do *patch* descreve o erro, a sua correcção, assim como a descrição das condições de teste. Tal informação será utilizada posteriormente por outras organizações dentro do processo. Esta informação é introduzida através da Ferramenta *SSG management*.

A descrição do *patch* que deve ser clara e precisa, será mais tarde incorporada na base de dados SSG depois da respectiva revisão.

A geração de um *patch* produz um ficheiro de comando, que a equipa de desenvolvimento ou a equipa teste, dará entrada na base de dados SSG. Estes ficheiros de comando estão normalmente agrupados e administrados num determinado grupo de subsistema.

### 3.2.5 Actualização de comentários no FEKAT (4)

Antes do *patch* ser libertado, o desenvolvimento deve introduzir o comentário correspondente ao relatório de falha no FEKAT.

Versões de sistema com correcções na fonte devem também ser incluídas no FEKAT.

Depois do *patch* ter sido testado com sucesso pela equipa de desenvolvimento, é passada para estado '02' (ver nota) e reportado para a equipa do teste de sistema, com inserção de comentário no FEKAT.

### 3.2.6 Teste do patch pela equipa do teste de sistema (5)

O anúncio dum *patch* disponível em estado '02' para um determinado *Fault report* vai conduzir ao teste do mesmo pela equipa do teste de sistema, que, após o teste com sucesso, colocará em '04' (ver nota) o estado da *patch* na base de dados SSG. Seguir-se-á a respectiva inserção de comentário no FEKAT, fechando assim o relatório de falha. O teste do *patch*, quando negativo, originará novo comentário no FEKAT, sendo a equipa de desenvolvimento responsável pela nova alteração do mesmo.

**Nota:** Estado de *patches*

- 02 - *Patch* testado pela equipa do desenvolvimento
- 04 - *Patch* testado pela equipa do teste de sistema
- 08 - *Patch* errado

### 3.3 Tipos de *Patches*

A distinção de tipos de *patches* serve para facilitar o seu processamento.

#### 1) *Backout Patches*

Se os *patches* forem incorrectos, deverá ser possível retirá-los do software de aplicação (APS), utilizando para o efeito um *backout patch*.

*Backout patches* são unicamente gerados quando requeridos para eliminar outros.

#### 2) *Master and Slave patches*

*Slave patches* podem ser introduzidos directamente no APS.

*Patches* com nomes de módulos, variáveis e endereços numa forma simbólica são chamados *Master patches*. Estes podem ser utilizados em vários projectos, através da geração dos *Slave patches* (mudando apenas os endereços).

Está disponível na base de dados SSG uma ferramenta que permite automaticamente carregar estes *patches* convertendo o módulo simbólico e nomes de objectos num endereço físico.

#### 3) *Patches* com dependências

Na documentação de um *patch* está previsto um campo que deverá obrigatoriamente ser preenchido se houver alguma dependência que poderá ser :

- dependência funcional;
- dependência com o projecto;
- dependência de procedimento de incorporação;
- dependência de HW e SW.
- dependência de documentação de cliente.

Um pacote físico de *patches*, contendo um ou vários *patches*, deverá ser definido se a correcção de um erro necessitar de vários *patches* ou se estes forem incorporados numa sequência específica.

## 4. Conclusões

Este artigo pretende evidenciar a forma sistemática do tratamento dos erros de software desde a sua detecção até à implementação da respectiva correcção.

O acompanhamento e registo das várias fases do processo permitem uma rastreabilidade completa incluindo as implementações no campo.

## Abreviaturas e definições

APS	Application Program Software - software de aplicação
EWSD	Central electrónica de Comutação Digital da Siemens
Fault Report	Relatório de falha
FMERF/PROCONS	Base de dados para registo e tratamento local dos relatórios de falha
FEKAT	Base de dados centralizada (a nível mundial) para controlo dos relatórios de falha
CM	Configuration Management - Base de dados para controlo da configuração do software (módulos, sub-sistemas, sistemas)
SSG	Base de dados de Grupos de Sub-Sistemas
Patch	Alteração de software introduzida ao nível do código máquina

## Bibliografia

1. [Siemens] Manual do Após-Venda
2. [Siemens] Manual do FEKAT
3. [Siemens] Manual do PROCONS
4. [Siemens] Manual SSG Management
5. [Rydin 95] Rydin, Carlos; Cortiço, Odete e Patrão, João  
“Gestão de Configurações de Software de Telecomunicações”  
Actas do Quatic 95, Lisboa, Dezembro 1995
6. [Almeida 95] Almeida, Leonor; Nascimento, Nuno e Pinto, Luís  
“SEPP<sup>©1</sup>: O Processo de Desenvolvimento, Produção e Manutenção de Software para Sistemas de Telecomunicações”  
Actas do Quatic 95, Lisboa, Dezembro 1995

# SIEMENS

Departamento de Comutação e Software.

QUATIC '95

2º Encontro Nacional para a Qualidade nas  
Tecnologias de Informação e

Comunicações.

LNEC, Lisboa, 4-6.12.95

**Curriculum Vitae abreviado dos autores da Comunicação:**

**“Processo de introdução de melhorias e de correcções de erros de Software em Sistemas de Telecomunicações”.**

**Engº. José Luis Sousa Oliveira**

Nascimento: Lisboa, 27.04.42

Grau académico: Licenciatura em Engenharia Electrotécnica pelo IST, em 1965

Dados Profissionais: Grupo Centrel: 1969-1987

Emptel/Siemens: 1987-1992

Desde 1993: Chefe de divisão, responsável pela Qualidade da Divisão de Vendas e Serviços do Departamento de Comutação e Software da Siemens S.A.

**Engº. Miguel Prudêncio**

Nascimento: Portimão, 26.10.59

Grau académico: Bacharel em Engenharia Electrónica e Telecomunicações, pelo ISEL, em 1983.

Dados Profissionais: Estágio na Siemens em 1983.

Ingresso na Siemens em 1984, função: coordenador da área de teste.

Ingresso na Timex (Divisão Computadores) em 1986, função: Desenvolvimento de Hardware.

Ingresso na Emptel/Siemens em Abril'87 para a Divisão de Vendas e Serviços.

Desde Outubro de 1991: chefe de sector da área de teste de sistemas.